Concept similarity measures the understanding between two agents

Jesus M. Olivares-Ceja, Adolfo Guzman-Arenas

Centro de Investigacion en Computacion (CIC), Instituto Politecnico Nacional, Mexico
a.guzman@acm.org

Abstract. When knowledge in each agent is represented by an ontology of concepts and relations, concept communication can not be fulfilled through exchanging concepts (ontology nodes). Instead, agents try to communicate with each other through a common language, which is often ambiguous (such as a natural language), to share knowledge. This ambiguous language, and the different concepts they master, give rise to *imperfect* understanding among them: How well concepts in ontology O_A map[1] to which of O_B? Using a method *sim* that finds the *most similar concept* in O_B corresponding to another concept in O_A, we present two algorithms, one to measure the similarity between both concepts; another to gauge **du**, the *degree of understanding* that agent A has about B's ontology. The procedures use word comparison, since no agent can measure **du** directly. Method *sim* is also compared with *conf,* a method that finds the *confusion* among words in a hierarchy. Examples follow.

1. Introduction and objectives

The easiest thing for two agents seeking communication (information exchange) is agreeing first in *what to communicate,* how and in what order, and then, doing it. Unfortunately, this requires a "wiser creature" (a programmer, or a Standards Committee) to establish these agreements.[2] In this paper **we will assume that no creature** of this kind is to be used. Then, what can an agent do to meaningfully communicate[3] with other agents (or persons), even when it/they had not made any very specific commitment to share a private ontology and communication protocol? Concept communication can not be fulfilled through direct exchange of concepts belonging to an ontology, since *they do not share* the same ontology, and O_A and O_B are in different *address spaces*. Instead, they should use a common language for communication. Lucky agents can agree on a language whose words have a *unique meaning.* Others need to use an ambiguous language (such as a natural language) to share knowledge. This gives rise to imperfect understanding and confusion.

In a different approach, [16] proposes to use natural language *words* as concepts.

---

[1] O_A and O_B are the ontologies of agents A and B, in the rest of this document.
[2] That is, the agents need to communicate *in order to agree* about how to communicate.
[3] Agent A communicates with B "in a meaningful way" when A moves towards its goals as the information exchange progresses. Each could be a person or a piece of software.

When one agent is talking to another, can the talker discover on what it (or the listener) is confounded? Is it possible to measure this mix-up? How can I be sure you *understand* me? Can I *measure* how much you understand me? Can *you* measure it? These questions have intrigued sociologists; they are also relevant to agents "not previously known to each other"[4] trying to "interact with free-will",[5] for which they have to exchange knowledge. The paper gives answers for them.

Knowledge is stored in concepts (Cf. §1.2), which are mapped by the talker into words of the communication language; perceived words are internalized as concepts by the listener. If the concepts exchanged are animals and plants, Latin is fine: *Felix Leo* represents the concept `lion-león-loin`[6] while *Cannabis Indica* stands for the concept `marijuana`. Other examples of words or symbols with a unique (universal) meaning: 4, π, Abelian group, Mexico, (23°22'57"N, 100°30'W), Abraham Lincoln, Berlin Symphony Orchestra. There are also semi-universal (popular) conventions [such as standard naming for chemical compounds, the Catalog for books of the Library of Congress, or the USA Social Security Number], which provide non-ambiguity for those who adhere. If two agents can select a non-ambiguous language (each of its words maps exactly to one concept) or convention to exchange concepts, great. Otherwise, they have to settle for an ambiguous language, such as English [7].

If two agents do not share a concept (figures 1 and 2), at least partially, they can not communicate it or about it. Thus, a measure of the amount of understanding can be the number of concepts they share, and *how well* they share them.[7] We will sharpen these measures for both cases: the ambiguous and the non ambiguous communication language.

---

[4] It is much easier to design the interaction (hence, the exchange of concepts) between two agents (each could be a person, or a piece of software), when the *same* designer designs both agents. In this sense, "they previously know each other:" each agent knows what the other expects, when, and the proper vocabulary to use. In contrast, our approach will allow *my* agents to interact with *yours,* as well as with Nigerian agents.

[5] Not-free will or canned interactions are those that follow defined paths. For instance, the interaction between a program and a subroutine it calls, where the calling sequence (of arguments) is known to both. Free will requires goals, resources, and planning [16].

[6] We represent concepts in `Courier font`. A concept is language-independent: the concept `cat` is the same as the concepts `gato-gata`, `gatto-gatta`, `chat-chatt`, `Katze`, `КОТ-КОШКА`, meaning "a small domestic feline animal." Concepts appear in English in this paper, for readers' benefit.

[7] Knowledge is also stored in the relations (or verbs, actions, processes) between objects (or nouns, subjects): A `balloon` can `explode`. It is also stored in the properties (or adjectives, adverbs) of these nouns and relations. The value of a property also contains information. A relation such as `explode` can also have relations (such as `father_of`, `speed_of_explosion`) which can also be nodes (concepts) in the ontology. Words are *not* nodes, but they are attached to the node they *denote*. The definition of ontology in §1.2 brings precision to these ideas: *a concept is always a node of an ontology* (and vice versa), whereas a relation and the other object related or linked (relations are arcs among two objects) may be a concept (node) or just a word or a token as defined in footnote 10. A relation that is just a token is called a property. If the "other object" is just a token, it is called a value. Examples: (`cat drinks milk`), (`balloon price` 50_cents), (`balloon` inflated_by `JFKennedy`), (`balloon` vanished languidly).

1.1 Related work

An ancestor of our *sim* (§3.1) matching mechanism is [3], based on the theory of analogy. Most work on ontologies involve the construction of a single ontology (for instance, [13]), even those that do collaborative design [10]. Often, ontologies are built for man-machine interaction [12] and not for machine-machine interaction. Work [2] identifies conceptually similar *documents* using a single ontology. Sidorov [4] does the same using a topic hierarchy: a kind of ontology. Linguists [14] identify related *words* (semantic relatedness), not *concepts*, often by statistical comparisons.

Huhns [11] seeks to communicate agents sharing a single ontology, such as O_D (Fig. 1). The authors are motivated [7] by the need of agents to communicate with unknown agents, so that not much *a priori* agreement between them is possible.[4]

```
thing (thing, something, object) {
   living (organism, life, being, creature, living thing) {
        animal (animal) {
                man (man, person, human being, woman, girl)
                        [eats = apple, peach ] }
        plant (plant, vegetal) }
   inanimate (inanimate object, tangible object) {
        rock (rock, stone)
        food (food, foodstuff, provisions) {
                solid_food (solid food){
                        apple (apple)[shape = round][color = red, yellow, green]
                        peach (peach) [color = orange, green, yellow]
                                      [shape = round]
                        bread (bread) [color = brown] }
                liquid_food (liquid, drink) {
                        water (water)
                        coffee (coffee, coffee drink, espresso) [color = black]
                        milk (milk) [color = white]
                        beer (beer) wine (wine) } } }
   abstract_thing (intangible object, abstract object, abstract thing) }
```

Ontology O_D

Fig. 1. *Ontology O_D: Foods.* Concepts appear in `courier font`, words denoting a concept are in (parentheses). Properties are inside [ ], such as [color = red, yellow, green]

Simple measurements [9] between *qualitative values* ("words") belonging to a *hierarchy*, find out how close two values are: the *confusion* between these values is measured. More at §3.1. Also, there is much work on tree distances.

With respect to the communication language, we prefer, in decreasing order:

1. A language whose tokens (words, atoms) are formed by concepts [8];
2. One with unambiguous tokens (a token represents only one concept). Examples: the Natural Numbers; the Proper Nouns;
3. One where each token has a small number of ambiguities, for instance, a natural language [14];

4. One where each token points to a foreign address space, thus representing a black box that can only be compared with = and ≠. Example: a language consisting of tokens such as "brillig", "toves," "borogove," "mome", "outgrabe", "fromelic," "meratroping"...

Approaches and experimental results through semantic analysis are in book [1].

1.2 Definitions

Level of a node in a tree. The root has level 0. A node has level ℓ+1 if his father has level ℓ.♦ [8]

Concept. An object, relation, property, action, process, idea, entity or thing that has a name: a word(s) in a natural language. ♦ Examples: `peak-uttermost`, `angry-mad`, `to_fly_in_air`. So, *concepts have names*: those words (or word phrases, such as New York City) used to denote them. A concept is unambiguous, by definition.[6] Unfortunately, the names given by different people to concepts differ and, more unluckily, the same word is given to two concepts (examples: words peak; fly; mad). Thus, *words are ambiguous*, *while concepts are not.* A person or agent, when receiving words from a speaker, has to solve their ambiguity in order to understand the speaker, by mapping the words to the "right" concept in his/her/its own ontology. The mapping of words to concepts is called *disambiguation*.

There are also composite or complex concepts, such as "to ski in a gently slope under a fair breeze while holding in the left hand a can of beer." These can be shared with other agents, too, but they do not possess a name: they have not been reified.

Ontology. It is a formal explicit specification of a shared conceptualization [5].♦ It is a taxonomy of the concepts we know.[9] We represent an ontology as a graph where each node is a concept and the arcs are relations to other concepts or tokens. [10] Some relations are concepts (such as `subset`, `member_of`, `part_of`, `eats-ingests`, `lives_in`); others are just tokens (which are called properties), represented in Times font, such as "color." Each relation links a node with other node (a concept) or with a token,[10] in the last case the token is called the value of the relation; for instance, "blue." In addition, associated to each node are the words that represent or denote that concept. The relation `subset` is represented by { }. §5 suggests a better representation. Examples: O_D and O_T in figures 1 and 2.

Size of O_A. Written as $|O_A|$, is the number of concepts in O_A. ♦

Teaching and learning. Agent T *teaches* agent S, and S *learns* from T, a set of concepts that T knows, if T patiently (incrementally) sends appropriate trios [of the form (concept relation concept)] to S such that S can build new nodes on its ontology, resembling those nodes already present in T. ♦ Agent T must often query S to see if "it has learned right", and to resolve contradictions or questions from S arising from its previous knowledge. More at §3.3.

---

[8] Symbol ♦ means: end of definition. Having a name in a shared language means that it is known to many people.

[9] Each concept that I know and has a name is shared, since it was named by somebody else.

[10] These tokens are words or strings of the types 2, 3, 4 of the list at the end of §1.1.

2. Measuring the amount of knowledge

How much does an agent know?

The amount of knowledge an agent has, is the number of concepts in its ontology. ♦ It is $|O_A|$. It is the area under the histogram of concepts (see Clasitex in [6]). This definition will be revisited in §3. *To know* a given discipline is to possess many concepts from that discipline.♦

How much does an agent know of the knowledge possessed by other agent? By comparing their histograms of concepts, we can find out that A knows twice more concepts than B about Numismatics, and that A and B know the same number of concepts about dinosaurs. A more accurate measure is given in the next section, where it is called the *degree of understanding.*

3. Measuring the degree of understanding

Two definitions are needed to quantify the (imperfect) grasp of a concept by an agent. One is *the most similar concept* in O_B to concept c_A in O_A; the other is *the degree of understanding* of B about the knowledge of A, which A keeps in O_A.

Assume that A knows that a `diprotodon`$_A$[11] is a mammal of the Tertiary Age, but B knows that a `diprotodon`$_B$ is a bear-like animal, of prehistory, has fur, two long milk teeth, and a size 5 meters long and 2 meters tall. All these concepts can be perfectly represented with the tools of §§1-2, since the trio `(diprotodon-fossil`[12] `skin-epidermis fur-hair`) is present (and true) in O_B and absent in O_A, and similarly for the other relations and concepts. Each concept and each trio of O_A is known by agent A, and the same is true for O_B and B. But it is also advantageous to "concentrate on the nouns" and to say that `diprotodon` is vaguely or less known to A than to B, since `diprotodon`$_B$ has more properties and more relations in O_B than `diprotodon`$_A$ in O_A.

[11] We use sub index A to stress the fact that `diprotodon`$_A$ belongs to O_A.

[12] When explaining concepts to the reader (of this paper) through English words, the convention in [15] is good: we use the word for the concept followed by a dash followed by the word that represents the father of the concept. This provokes little ambiguity in the reader, and it has been programmed in *sim*, the mapper of a concept to the closest concept in another ontology (§3.1). Thus, we write `star-person, star-animal, star-astronomic_body, star-adornment,` for the four meanings of word star.

```
thing (thing, something, object, entity) {
  physical_object (concrete object, physical object) {
    living_creature (creature, life form, live being, living creature, organism, being){
      animal (animal) {
        invertebrate (invertebrate) {
          insect (insect, bug) {
            fly_animal(fly, flies) cockroach (cockroach) flea(flea) }
          mollusk (mollusk) }
        vertebrate (vertebrate) {
          reptile (reptile) {lizard (lizard)   iguana (iguana) }
          batrachians (batrachians) {frog (frog) }
          mammal (mammal) { zebra (zebra)
            rodent (rodent) {
              rat (mouse, mice, rat) {
                domestic_rat (domestic rat)
                country_rat (country rat) }
              mole-rodent (mole) }
            fox (fox) cat(cat, kitty) dog(dog) lion(lion, cub) donkey(donkey)
            man (man, men, woman, women, person, people, human being,
                   Homo Sapiens, boy, girl, child, miss, mister, sir)
              [eats = tropical_plant, citrus] }
          bird (bird) { chicken (chicken, hen, cock, rooster, chick, poultry)
            duck (duck)  parrot (parrot)  hawk (hawk) }
          fish (fish) } }
      plant-creature (plant, vegetal) {
        tropical_fruit (tropical fruit) {
          coconut (coconut) mango (mango, mangoes) }
        citrus (citrus, citric) { lemon (lemon) orange (orange)
            tangerine (tangerine) [color = orange]
            rare_fruit [color = green] [texture = smooth] [size = 5cm] }}
      bacteria (bacteria, microorganism) }
    artificial_object (artifact, artificial object) }
  abstract_object (imaginary object, abstract thing, abstract concept) }
```

Ontology O_T

Fig. 2. Ontology O_T. Properties are of the form [property-name = value], where the property name or the value may be concepts as in [eats = tropical_plant, citrus] or tokens of the types of footnote 10, as in [color = orange].

<u>Definition.</u> The *degree of knowledge* of A about a concept c is a number between 0 and 1, obtained by counting the number of relations containing c in O_A, adding the number of properties of c in O_A, and dividing into the similar calculation for c in the total ontology.[13] ♦ The closer it is to 1, the less *imperfect* is A's knowledge of c. ♦ This definition is impractical to use since the total ontology is out of our reach. Thus, we shall compute instead the degree of knowledge of an agent *with respect to another agent*, which we refer to in §3.2 as the *degree of understanding* of A about O_B: how much A understands about what B knows; how well each concept of A maps into the

[13] The ontology of an agent that knows much, if not everything.

corresponding (most similar) concept in B. Our examples refer to figure 2. First, we need to find the concept in O_B most similar to one given (through words describing it) by A, belonging of course to O_A.

3.1 Finding the concept in O_B most similar to a given concept in O_A

Algorithm *sim* [8] (called "*hallar*(c_A)" or COM in [15]) finds *the most similar concept* c_B in O_B to concept c_A in O_A. Agent A makes known concept c_A to B by sending to B words[14] denoting c_A, and also sending words denoting c_A's father. Also, *sim* returns a similarity value $sv \in [0, 1]$ expressing how similar was c_B to c_A.

If c_B is the concept most similar to c_A, it is not necessarily true that c_A is the concept most similar to c_B. *Function sim is not symmetric.* Example: A physician P knows six kinds of `hepatitis`, including the popular hepatitis type A, while John only knows `hepatitis`. Each of the six `hepatitis` of P finds John's `hepatitis` as "the most similar concept John has," while John's `hepatitis` best maps into P's `type_A_hepatitis`. P knows more than John, so P can select a better target in his rich ontology for John's vague concept. *John can not make such selection.*

The function *sim* is only defined between a concept c_A in O_A and *the most similar concept* c_B in O_B. Extensions sim' and sim'' appear below.

Who runs sim? Who compares these two concepts, since they belong to different ontologies? That is, who runs *sim*? Either agent A or B can execute it, since *sim* compares words, not concepts. But, when A runs *sim*, it needs the collaboration of B (and vice versa), which has to provide words to be used by *sim* (thus, by A). Also, even when A executes *sim* producing c_B as result, A can not "have" or "see" c_B: it is a pointer to the memory O_B, a meaningless pointer for A, such as the tokens of point 4 of §1.1. The most of what A can see of c_B is *(1)* the words which denote c_B, as well as (the words for) the relations of c_B; *(2)* corresponding words for the father, grandfather, sons... of c_B (and words for *their* relations); *(3)* value *sv*, indicating how similar that elusive c_B is to its (very solid) c_A. In fact, A still has c_A as "the concept I have been thinking all along." When B runs *sim,* B can see, of course, c_B, but it can not "see" or "grasp" c_A. The most of what B can see of c_A is that "agent A wants to talk about something of which the closest I have is c_B".[15] B can sense from the words sent to it by A differences between its solid c_B and the elusive c_A of A. More in §3.3.

---

[14] By §1, A can not send any *node* of O_A to B. If later in the algorithm, A needs to send a relation of c_A to B (such as `color`), it sends the words (color, hue) corresponding to such relation `color`. No concepts travel from A to B or vice versa, just words denoting them.

[15] It will not help if A is more cooperative. For instance, dumping all its O_A into B's memory will not help B, who will still see a tangled mesh of meaningless pointers. Well, not totally meaningless –some understandable words are attached to each node (concept). Yes: B can slowly understand (untangle) O_A by comparing each concept in O_A with every concept in its own O_B –that is, by using *sim*! See §5 "Suggestions for further work."

Generalizing sim. Function sim'(c_A, d_A) for two concepts belonging to the *same* ontology, is defined as 1/(1+length of the path going from c_A to d_A in the O_A tree). ♦ The path is through `subset` and `superset` relations; its length is the number of such arcs traveled. sim'(c_A, d_A) $\in$ (0, 1]. sim' is symmetric. Example: Figure 3.

Relation to confusion. In [9], the confusion conf(c_A, d_A) occurring by using c_A instead of d_A, is defined as the length of the descending[16] path from c_A to d_A. ♦ This definition holds for hierarchies; it is here extended to ontologies. If we had defined sim'(c_A, d_A) = (1/(1 + length of the descending path going from c_A to d_A in the O_A tree), we would have had sim'(c_A, d_A) = 1/1+conf(c_A, d_A). We prefer, for ontologies, the first definition of sim', since it is symmetric, while conf is not. Example: for ontology O_D of figure 1, conf(`liquid_food, food`) = 0; the confusion when using `liquid_food` instead of `food` is 0, since liquid food is food. But conf(`food, liquid_food`) = 1; when I want liquid food but I am given food, there is an error of 1 (a small error, you could think). More in figure 3.

Confusion and similarity for concepts x and y belonging to the same ontology O_D of figure 1	conf(x, y); confusion in using x instead of y	conf(y, x); confusion in using y instead of x	sim' (x,y) = sim' (y,x); similarity between x and y
`x = man, y = living`	0	2	1/3
`x = peach, y = bread`	1	1	1/3
`x = bread, y = water`	2	2	1/5
`x = water, y = man`	3	4	1/8
`x = bread, y = coffee`	2	2	1/5
`x=thing, y=wine`	4	0	1/5

Fig. 3. Examples of confusion and similarity (sim') for two concepts of the same ontology (O_D, figure 1). Function *conf* is not symmetric; *sim'* is.

For similarity between *any* two objects in different ontologies, we have:

sim''(c_A, d_B) is found by making first s_1 = *sv* returned by sim(c_A) [this also finds c_B, the object in O_B *most* similar to c_A]; then, find s_2 = sim'(d_B, c_B). Finally, sim''(c_A, d_B) = $s_1 s_2$. ♦

[16] Going towards more specialized concepts. Using a person from Dallas when I want to use a Texan person, confusion is 0; using a Texan person when a Dallas person is needed causes confusion=1; using a US person causes confusion=2.

3.2 Degree of understanding

The value *sv* found in c_B = sim(c_A) in §3.1 can be thought of as the degree of understanding that agent B has about concept c_A. A concept c_A that produces *sv*=0 indicates that B has understanding 0 (no understanding) about that c_A. Averaging these *sv's* for all concepts in O_A, gives the degree of understanding that agent B has about the whole ontology O_A of A.[17] It is as if agent A examines and asks B, for each concept $c_A \in O_A$, «Do you understand what is c_A?» «How much do you understand c_A?» At the end, A and B have a good idea of the understanding of B (about O_A).

The *degree of understanding* of B about O_A, du(B, O_A) = {sum over all $c_A \in O_A$ of *sv* returned by sim(c_A)} / $|O_A|$. ♦ It is the average of the *sv*'s. Similarly, we can measure the degree of understanding of B about *some region* of O_A. ♦ Function *du is not symmetric.* In general, an agent understands some regions better than others. If $|O_A| >> |O_B|$, then du(B, O_A) is small: B knows little about O_A, even if all parts of O_A known to B were to have *sv*=1.

du(B, O_A) ≤ 1; in regions where B knows more than A, du = 1. Example: assume agent T has the ontology O_T of figure 2 and agent N has ontology O_N of figure 4. Then, figure 5 shows the concept c_T most similar to each $c_N \in O_N$, as well as the corresponding similarity value *sv.* Thus, du(T, O_N) = (Σ *sv*)/$|O_N|$ = 9.58/15 = 0.64 is the degree of understanding that agent T has about ontology O_N.

O_N

```
living_creature (organism, being, living creature) {
        animal (animal)
                frog (frog, tadpole)   iguana (iguana)
                diprotodon (diprotodon) }
        plant-creature (plant, vegetal) {
                big_plant (big plant, large plant) {
                        coconut (coconut)   mango (mango) }
                small_plant (small plant) {
                        strawberry (strawberry)   lemon (lemon) } }
        bacteria (bacteria) {
                Escherichia_Coli (Escherichia Coli, E. Coli)
                Streptococus_aureus (Streptococus Aureus, S. Aureus)} }
```

Fig. 4. Ontology O_N. The degree of understanding that agent T has about O_N is 0.64 (Fig. 5)

On the other hand (figure 6), to find out the degree of understanding that agent N (with ontology O_N) has about ontology O_T of figure 2, we need to find sim(c_T) for each $c_T \in O_T$, and to average their *sv*'s. Thus, du(N, O_T) = (Σ *sv*)/$|O_T|$ = 10.08/47 = 0.21 is the degree of understanding that agent N has about ontology O_T. N knows less about O_T than T about O_N. The *understanding* of O_N by T increases as each *sv*

[17] B does not know how many concepts there are in O_A, so it needs cooperation from A, for instance, when B asks A "give me the next concept from your ontology."

increases and O_T grows. In more realistic ontologies, *relations* (such as `eats, part_of, lives_in`) are also nodes of the ontology, contributing to *du*.

c_N	c_T = sim (c_N)	*sv* of sim
living-creature	living_creature	0.8
animal	animal	1
plant_creature	plant_creature	1
bacteria	bacteria	1
frog	frog	0.64
iguana	iguana	0.64
diprotodon	son_of animal	0.5
big_plant	son_of plant_creature	0.5
small_plant	son_of plant_creature	0.5
coconut	coconut	1
Escherichia Coli	-	0
Streptococus Aureus	-	0
mango	mango	1
lemon	lemon	1
strawbery	-	0

Fig. 5. *How well T knows O_N.* Computing du(T, O_N) is like asking T how much it knows about each concept c_N. The *sv*'s in the last column are the answer. Adding these *sv*'s and dividing into $|O_N|$=15, the degree of understanding of T with respect to O_N is found to be 0.64

3.3 Finding and correcting the source of a disagreement

§3.1 shows that agent A can not perceive or see c_B directly. Given $c_A \in O_A$ and its most similar concept $c_B \in O_B$, can A perceive in what way c_B differs from its c_A? After all, A knows from the value *sv* returned by sim(c_A), how imperfect is the matching of c_B to c_A.

The answer is *yes*, and the following **Process P** computes it. Agent A can ask B about the relations in which c_B takes part [That is, arcs linking c_B with other concepts or with words or tokens[10]]. It will receive the answers in words. Then, A can process them (through *sim*) to see how c_A's relations differ from those received. It can do the same with the father_of(c_B), and with the sons_of(c_B). And so on. Some words received will refer to relations of which A is not sure (it has no name for them, or there is ambiguity), so that more processing (Process P is called again) *on these relations* is needed. Sometimes, B will mention (words for) a concept in O_B of which A is not sure (so, Process P is called again) or is not in O_A. Occasionally agent A will receive from B assertions about c_B which A has as false for c_A.

An agent learning from another agent. For concept c_A, agent A can make a note N_{cA} containing what B knows about c_A which differs from what A knows about c_A: the values c_B, *sv*, and other differences between c_A and c_B. N_{cA} could be considered a belief about agent B: B believes N_{cA} about c_A, and N_{cA} is not what A knows about c_A. As one step further, agent A can *internalize* N_{cA}; that is, own (believe, digest) N_{cA}: agent A can *learn* N_{cA} about c_A from O_B. For this to happen, agent A needs to incorporate the new relations and concepts (in N_{cA}) into its O_A, and to *resolve the ambiguities and inconsistencies* coming from N_{cA} (some of N_{cA}'s trios are known to A to be false [there is a contradiction]; others are ambiguous to A). This has been solved for an *agent* teaching a *person* but not yet for an agent teaching another agent. We have no solution now. It can be done, we think, by using other *knowledge services* in the Web to referee disagreements between O_A and O_B and help A decide *who is wrong* about what (the "what" is already captured in N_{cA}).

c_T	c_N = sim (c_T)	*sv* of sim
living_creature	living_creature	0.8
animal	animal	1
invertebrate	son_of animal	0.5
vertebrate	son_of animal	0.5
Iguana	iguana	0.64
frog	frog	0.64
plant_creature	plant_creature	1
tropical_fruit	son_of plant_creature	0.5
coconut	coconut	1
mango	mango	1
citrus	son_of plant_creature	0.5
lemon	lemon	1
bacteria	bacteria	1

Fig. 6. *How well N knows each concept c_T in ontology O_T?* Each answer (c_N, second column) yields a similarity value (last column); only *sv*'s ≠ 0 are shown. The following concepts in O_T found no similar concept (*sv*=0) in O_N (N does not know them): thing, physical_object, insect, fly_animal, cockroach, flea, mollusk, reptile, lizard, batrachians, mammal, zebra, rodent, rat, domestic_rat, country_rat, mole_rodent, fox, cat, dog, lion, donkey, man, bird, chicken, duck, parrot, hawk, fish, orange, tangerine, rare_fruit, artificial_object, abstract_object

4 Conclusions

• Methods are given to allow interaction and understanding between agents with different ontologies, so that there is no need to agree first on a standard set of concept

definitions. Given a concept and associated words, a procedure for finding the most similar concept in another ontology is shown, with examples, as well as a measure of the degree of understanding between two agents. It remains to test our methods with large, vastly different, or practical ontologies.
• Work exposed is a step towards free will interactions among agents, perhaps strange to each other (footnote 4), needing to make sense of their utterances. It opposes current trends: *(1)* using canned interactions (footnote 5), *(2)* using agents written by the same group, or *(3)* following the same data exchange standards.
• Interaction through standards (trend *3* above) will dominate the market for some time. A standard ontology in a discipline is a good thing, although it feels rigid and archaic after a while.[18] It is easier to follow standards than to be "flexible, uncompromising and willing to try to understand new concepts." Irrespective of standardization, our approach allows agents to be flexible and have general ways of trying to understand what each has to say, specially new or unusual things.
• *A standard ontology for concept-sharing is not needed;* if one is built, it will always lag behind, since new concepts appearing every day will not be in it.

5 Suggestions for further work

Machine learning. Do the internalization of N_{cA} mentioned in §3.3; then, generalize to each $c_A \in O_A$ and somehow (we do not know how now) to each $c_B \in O_B$. This will allow agent A *to learn* (without human help) O_B from B. The new O_A = its old O_A merged with O_B. Alma-Delia Cuevas works along these lines.
Ontology merging. Is there a faster way for A to learn O_B in §3.3? Surely, agent A can get rid of its O_A and use (copy) O_B instead. This is too drastic: agent A forgets or erases what it already knows, in favor of B's knowledge. Perhaps A can build O_B *on top* of A, and patch somehow O_A to accommodate for inconsistencies. *Suggestion:* use notes N_{cA}. This we have called *ontology merging*; more work is needed. Or there is the proposal by [16] to use *words* as concepts. *Improvement:* Add a crawler that combs the Web for ontologies suitable for merging.
Better notation for ontologies. • Tree notation is cumbersome, since only one `subset` relation is represented, and often a set S is partitioned into several partitions. Thus, a better notation could be:

```
person      partition sex (=M : male_person) (=F : female_person)}
            {partition age  (≤20 : young_person)
                            (20< age ≤ 50 : adult_person)
                            (>50 : old_person) }
```

• Similarly, graphs are cumbersome for representing n-ary relations. • When characterizing the relations (as another sub-tree of the ontology), you need to define types of partitioning relations (`sex`, `age`…), or whether the partition is a "natural"

[18] Compare the UNESCO Catalog of Sciences (which is 30-years obsolete in Computer Science) with the ACM Computing Classification System, which is 2-years obsolete.

one, like partitioning `vertebrate` into `fish`, `bird`, `reptile`, `batrachian` and `mammal`.
Agent interaction. Establish necessary or sufficient conditions for agent interaction lacking a communication agreement, as mentioned in §1.

Acknowledgments. Helpful discussions were held with Profs. Serguei Levachkine (CIC), Michael N. Huhns (U. of South Carolina), Alexander Gelbukh (CIC), and Hal Berghel (U. of Nevada Las Vegas). Work herein reported was partially supported by NSF-CONACYT Grant 32973-A and SNI-CONACYT *National Scientist* awards.

References

1. Victor Alexandrov, S Levachkine, Adolfo Guzman-Arenas. *"Data Dynamical Structures for Image Treatment with Applications to Digital Cartography"*. Book in preparation.
2. John Everett, D Bobrow, *et al* (2002) Making ontologies work for resolving redundancies across documents. *Comm. ACM* **45,** 2, 55-60. February.
3. K. Forbus, B. Falkenhainer, D. Gentner. (1989) The structure mapping engine: algorithms and examples. *Artificial Intelligence* **41***,* 1, 1-63.
4. A. Gelbukh, G. Sidorov, and A. Guzman-Arenas. (1999) Document comparison with a weighted topic hierarchy. *DEXA-99, 10th International Conference on Database and Expert System applications,* Workshop on Document Analysis and Understanding for Document Databases, 566-570. Florence, Italy, August 30 to September 3.
5. T. Gruber. A translation approach to portable ontologies. *Knowledge* **7**.
6. Adolfo Guzman-Arenas (1998) Finding the main themes in a Spanish document. *Journal Expert Systems with Applications,* Vol. **14,** No. 1/2, 139-148, Jan./Feb. Elsevier.
7. A. Guzman-Arenas, C. Dominguez, and J. Olivares. (2002) Reacting to unexpected events and communicating in spite of mixed ontologies. *Lecture Notes in Artificial Intelligence* **2313**, 377-386, Carlos A. Coello *et al* (eds), Springer Verlag, Heidelberg.
8. Adolfo Guzman-Arenas and Jesus Olivares. (2004) Finding the Most Similar Concepts in two Different Ontologies. Accepted in *MICAI 04.* To appear as part of a *LNCS* book.
9. A. Guzman-Arenas and Serguei Levachkine. (2004) Hierarchies Measuring Qualitative Variables. *Lecture Notes in Computer Science* **2945**, 262-274, A. Gelbukh (ed).
10. Cloyde W. Holsapple and K. D. Joshi. (2002) A collaborative approach to ontology design. *Comm. ACM* **45,** 2, 42-47. February.
11. M. N. Huhns; M. P. Singh. and T. Ksiezyk (1997) Global Information Management Via Local Autonomous Agents. In: M. N. Huhns, Munindar P. Singh, (eds.): *Readings in Agents,* Morgan Kauffmann Publishers, Inc. San Francisco, CA
12. Henry Kim. (2002) Predicting how ontologies for the semantic web will evolve. *Comm. ACM* **45,** 2, 48-54. February.
13. Douglas B. Lenat, R. V. Guha, Karen Pittman, Dexter Pratt and Mary Shepherd (1990). Cyc: Toward Programs with Common Sense, *Comm. of the ACM* **33**, 9, 30 – 49.
14. M. Montes-y-Gomez, A. Lopez-Lopez, and A. Gelbukh. (2000) Information Retrieval with Conceptual Graph Matching. *Lecture Notes in Computer Science* **1873**, 312-321.
15. Jesus Olivares (2002) *An Interaction Model among Purposeful Agents, Mixed Ontologies and Unexpected Events.* Ph. D. Thesis, CIC-IPN. Mexico (In Spanish) Available on line at http://www.jesusolivares.com/interaction/publica
16. Yorick Wilks, B. Slator, L. Guthrie. (1996) *Electric words. Dictionaries, computers, and meanings.* ACL-MIT Press. Cambridge, USA. ISBN 0-262-23182-4 (hc)